IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Devisree U V

16-09-2022

Outline

- Executive Summary [Slide]
- Introduction [Slide]
- Methodology [Slide]
- Results [Slide]
- Conclusion [Slide]
- Appendix [Slide]

Executive Summary

Problem Statement:

In the domain of rocket launches, the reusability of the rockets after the first stage plays a crucial role in determining the overall cost of the project. We aim to create competition against SpaceX company, one of the most successful companies in this domain, by offering a solution at a better cost. To summarize, we ask the following question: *Can we bid against SpaceX by predicting the success rate of rocket landing at the first stage?*

Methodology:

1. Data Collection from SpaceX Falcon 9 launches
    - API requests and Web Scraping -> Minimizes cost
2. Exploratory Data Analysis -> SQL, Data Visualization & Visualization with Folium
3. Predictive Modelling -> Machine Learning algorithms

Result and Observations:

We can predict the success rate of the rocket launches with a very high probability. This answer the problem statement that a company can bid against SpaceX safely. The study promises the use of machine learning tools in minimizing costs and estimating success rates.

Future Directions:

- Limited amount of data was used for analysis -> Increase the amount of data for better predictions
- Using more powerful machine tools like graph neural networks for analysis

Introduction

- Humans have achieved huge success in Space exploration in the recent past.
- This motivated several startups to invest in the space exploration domain.
- According to CBInsights, 70% of startup companies fail within 20 months after the seed fundraising.
- SpaceX, a dominating company in space rockets, gets away from this issue with the help of reusable rockets.
    - Successfully land rocket after the first stage and reuse it for subsequent launches.
- To compete with SpaceX, a strategy for predicting the success probability of first-stage landing is crucial.
- Solution: Predictive modeling on data from SpaceX Falcon 9 rocket launches.

Section 1

Methodology

Methodology

1. Data collection methodology
   - ❖ The data was collected by making requests to SpaceX API and Web Scraping by collecting Falcon9 launching records on the Wikipedia page.
2. Perform data wrangling
   - ❖ One hot encoding and Web Scraping using BeautifulSoup
3. Perform exploratory data analysis (EDA) using visualization and SQL
4. Perform interactive visual analytics using Folium and Plotly Dash
5. Perform predictive analysis using classification models
   - ❖ Created Machine Learning Pipeline using various classification models for predicting if the first stage will land successfully or not.

Data Collection: SpaceX API

- Data Collection is the major part of the project because we use this data to train our machine learning models to make precise predictions.
- Collection of data was done by various methods such as SpaceX API and Web Scraping.
- Used requests.get() method for data from SpaceX API, then we decoded the response content as a Json content using .json() and turned into Pandas dataframe using .json_normalize().
- requests.get() method

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
response = requests.get(spacex_url)
```

- using .json() and .json_normalize()

```
# Use json_normalize meethod to convert the json result into a dataframe
space_data = pd.json_normalize(response.json())
```

Data Collection – SpaceX API

- Constructed our data frame, done basic data wrangling, and filled the missing values wherever required for cleaning the dataset.

- Converted our data frame into a CSV dataset, named SpaceX_Data Set 1 API

- **URL:** https://github.com/itzdevisree/IBM-Data-Science-Capstone-Project/blob/master/1.%20SpaceX_Data%20Collection%20API.ipynb

Finally lets construct our dataset using the data we have obtained. We we combir

```
launch_dict = {'FlightNumber': list(space_data['flight_number']),
'Date': list(space_data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

```
# Calculate the mean value of PayloadMass column
PayloadMass_mean = data_falcon9['PayloadMass'].mean()
# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'] = data_falcon9['PayloadMass'].replace(np.nan,PayloadMass_mean)
data_falcon9
```

```
# Create a data from launch_dict
data = pd.DataFrame(launch_dict)
print(data)
```

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

Data Collection: Web Scraping(Cont.)

- Web scraping was performed using BeautifulSoup() to get Falcon 9 launch records from the Wikipedia page :

URL: https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches

- Stored the launch records in html tables.
- Parsed the html data tables and converted into a dataframe.
- Exported it to a CSV dataset, named SpaceX_Data Set 2 Web Scraping.
- URL: https://github.com/itzdevisree/IBM-Data-Science-Capstone-Project/blob/master/2.%20SpaceX_Data%20Collection%20with%20Web%20Scraping.ipynb

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

```
# use requests.get() method with the provided static_url
html_data = requests.get(static_url)
# assign the response to a object
html_data.status_code
```

```
200
```

Create a `BeautifulSoup` object from the HTML `response`

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(html_data.text,'html.parser')
```

```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names
soup_ext = soup.find_all('th')
for row in range(len(soup_ext)):
    try:
        name = extract_column_from_header(soup_ext[row])
        if (name is not None and len(name) > 0):
            column_names.append(name)
    except:
        pass
```

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

```
df=pd.DataFrame(launch_dict)
```

Data Wrangling

- Performed Exploratory Data Analysis(EDA) in order to find patterns and determine the label for training supervised models.
  - True RTLS = successfully landed on the ground pad,
  - False RTLS = unsuccessfully landed on the ground pad,
  - True ASDS = successfully landed on a drone ship,
  - False ASDS = unsuccessfully landed on a drone ship.
- Calculated #launches of each site and #occurrences of each orbit.
- Created landing outcome labels from the outcome column and converted these into training labels – 1 for the successful landing and 0 for unsuccessful.
- Calculated the success rate of Booster Falcon 9 and exported the results to CSV dataset named, SpaceX_Data Set 3 Wrangling.
- **URL**: https://github.com/itzdevisree/IBM-Data-Science-Capstone-Project/blob/master/3.%20SpaceX_EDA%20Data%20Wrangling.ipynb

```
# landing_outcomes = values on Outcome column
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes
```

```
bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
bad_outcomes
```

```
{'False ASDS', 'False Ocean', 'False RTLS', 'None ASDS
```

TASK 4: Create a landing outcome label from Ou

Using the `Outcome`, create a list where the element is zero if the c
assign it to the variable `landing_class`:

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = []
for outcome in df['Outcome']:
    if outcome in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

```
df['Class']=landing_class
df[['Class']].head(8)
```

```
df["Class"].mean()
```

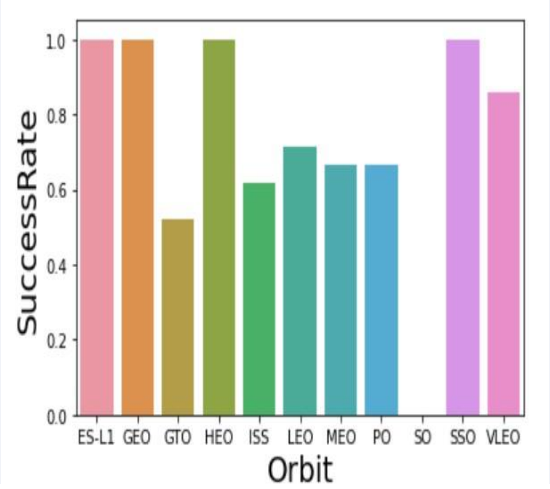
```
0.6666666666666666
```

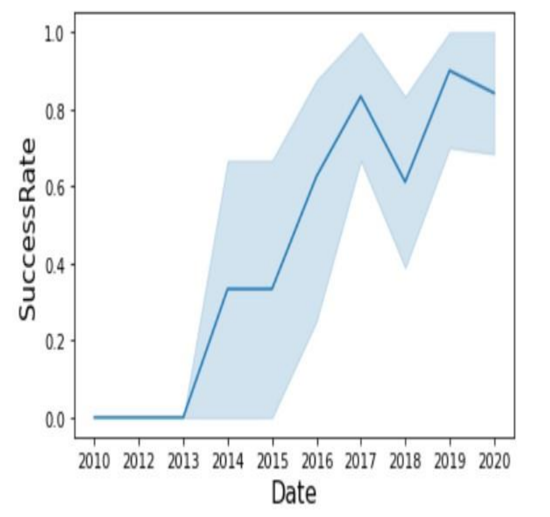
We can now export it to a CSV for the next section,but t

```
df.to_csv("dataset_part_2.csv", index=False)
```

EDA with Data Visualization

- Data Visualisation helps to understand data by using Line Charts, Scatter plots, Bar charts, etc to highlight our trends and outliers. We perform exploratory Data Analysis(EDA) and Feature Engineering using Pandas and Matplotlib.
- Explored by visualizing the relationship between flight number & payload, flight number & launch site, launch site & payload using Scatter plots.
- Visualized the relationship between success rate and orbit type using Bar chart & visualized the launch success yearly trend using a Line chart.
- Obtained some insights that affect success rate then One hot encoding was applied and the result exported as CSV dataset named SpaceX_Data Set 4 Visualisation.
- **URL**: https://github.com/itzdevisree/IBM-Data-Science-Capstone-Project/blob/master/5.%20SpaceX_EDA%20with%20Data%20Visualization.ipynb





EDA with SQL

- Loaded the dataset into the database to get insights from the data.
- Wrote SQL Queries for obtaining:
    - the names of the unique launch sites
    - the total payload mass carried by boosters launched by NASA (CRS)
    - average payload mass carried by booster version F9 v1.1
    - the total number of successful and failed mission outcomes
    - the names of the booster versions which have carried the maximum payload mass.
    - the failed landing outcomes in drone ship, their booster versions, and launch site names
    - the date when the first successful landing outcome in ground pad was achieved
    - the count of landing outcomes (such as Failure (drone ship) or Success (ground pad))
- **URL:** https://github.com/itzdevisree/IBM-Data-Science-Capstone-Project/blob/master/4.%20SpaceX_EDA%20with%20SQL.ipynb

Build an Interactive Map with Folium

- Folium Markers were used to mark all the SpaceX Launch Sites and to explore and analyze the proximities of launch sites to railways, coastline, highways, and cities.
- Folium Circles are used to highlight the launch site area.
- Polylines are used to connect launch sites and their nearest landmarks
- Assigned a feature named launching outcomes for marking Success or Failure to Class 0 and 1 respectively.
- Color-labeled marker clusters used where **GREEN** represents the success of the rocket launch site, **RED** represents rocket launch failure.
- **URL:** https://github.com/itzdevisree/IBM-Data-Science-Capstone-Project/blob/master/6.%20SpaceX_Data%20Visualisation%20with%20Folium.ipynb

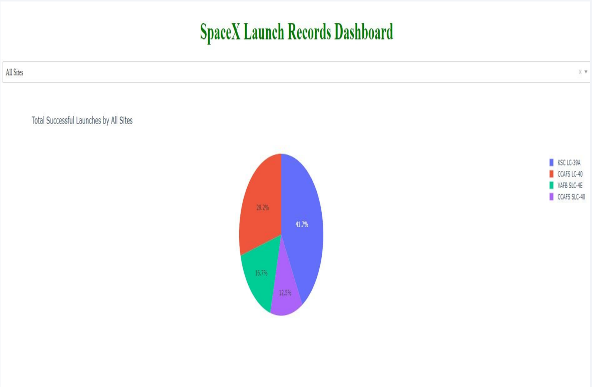
Launch Site	Lat	Long	class	marker_color
KSC LC-39A	28.573255	-80.646895	1	green
KSC LC-39A	28.573255	-80.646895	1	green
KSC LC-39A	28.573255	-80.646895	1	green
CCAFS SLC-40	28.563197	-80.576820	1	green
CCAFS SLC-40	28.563197	-80.576820	1	green
CCAFS SLC-40	28.563197	-80.576820	0	red
CCAFS SLC-40	28.563197	-80.576820	0	red
CCAFS SLC-40	28.563197	-80.576820	0	red

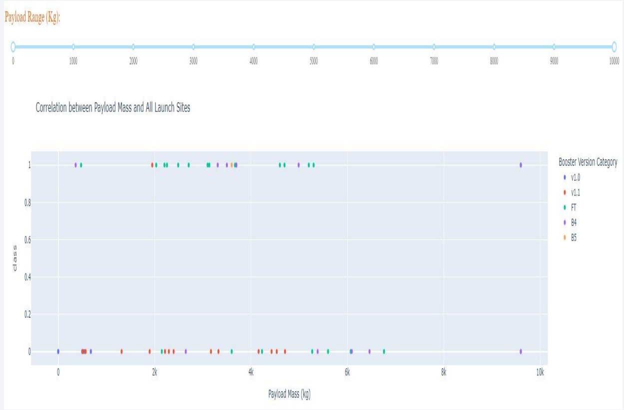
```python
# Apply a function to check the value of `class` column
# If class=1, marker_color value will be green
# If class=0, marker_color value will be red

launch_site = []

for i in enumerate(spacex_df['class']):
    if i==1:
        launch_site.append('Green')
    else:
        launch_site.append('Red')
```

```python
# Function to assign color to launch outcome
def assign_marker_color(launch_outcome):
    if launch_outcome == 1:
        return 'green'
    else:
        return 'red'

spacex_df['marker_color'] = spacex_df['class'].apply(assign_marker_color)
spacex_df.tail(10)
```

Build a Dashboard with Plotly Dash

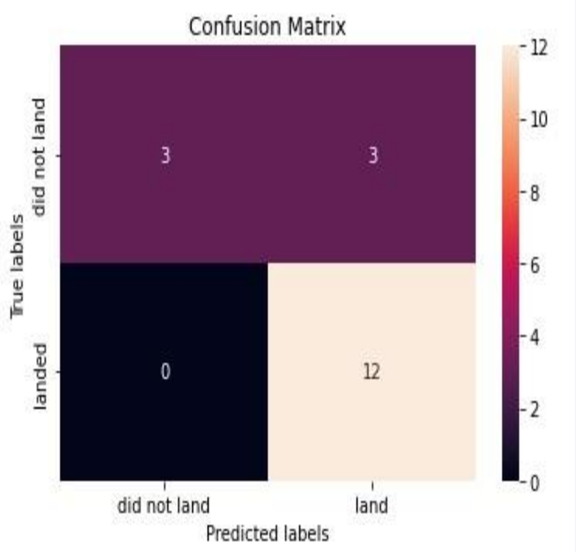
- We built an Interactive Dashboard with Plotly Dash.
- Pie Charts and Scatter plots are used to visualize the launch records of SpaceX.
- In the dashboard, Pie Charts visualized the success rates of all launch sites.
- Scatter plots visualized the correlation between the Payload mass and Launch site for different Booster versions.
- Charts helps to understand the factors that affect the success rates of launch sites such as Payload mass and Booster versions.
- **URL**: https://github.com/itzdevisree/IBM-Data-Science-Capstone-Project/blob/master/7.%20SpaceX_Plotly_Dash_App.ipynb

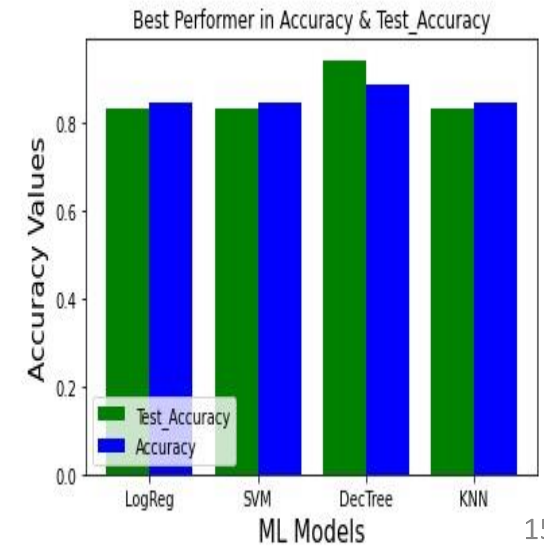




Predictive Analysis (Classification)

- Loaded the data using pandas and using NumPy converted into pandas series.
- Standardized and transformed the data, then split our data into training and testing.
- Built different machine learning models and calculated the best hyperparameters using GridSearchCV for predictions.
- Calculated the accuracy using Logistic Regression, Support Vector Machine(SVM), Decision Tree and K-Nearest Neighbour (KNN) methods and visualized the values using a confusion matrix.
- Compared the accuracy values of the training and testing set and visualized using barplot.
- Using the barplot, the method which is the best performer in the training and testing set is identified.
- **URL:** https://github.com/itzdevisree/IBM-Data-Science-Capstone-Project/blob/master/8.%20SpaceX_Machine%20Learning%20Prediction.ipynb





Results

➢ Exploratory Data Analysis :

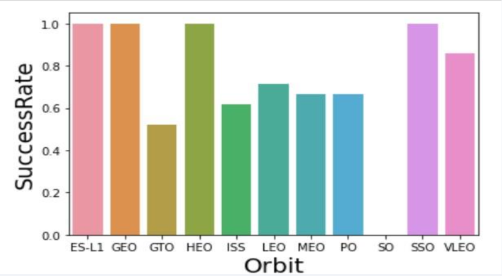
- the successful landing outcomes are related with the Booster version and Payload mass
- the successful landing outcome have significant increase since the year 2013.
- success rate of launch sites with heavy payloads with orbits SSO, GEO,HEO, ES-L1, etc is more.

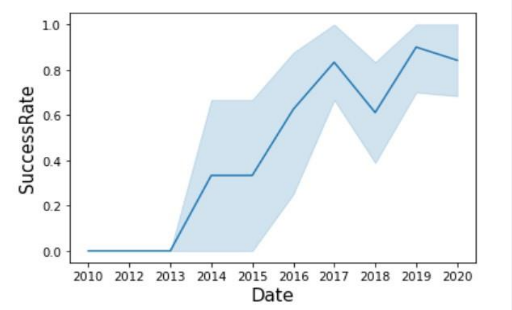
➢ Interactive Analytics:

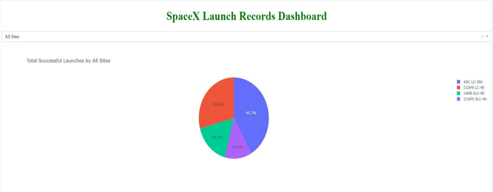
- the launch site KSC LC 39A has largest successful launches and high success launch success rate.
- the Booster version FT and the payload range between 2000kg and 4000kg have higher success rate.

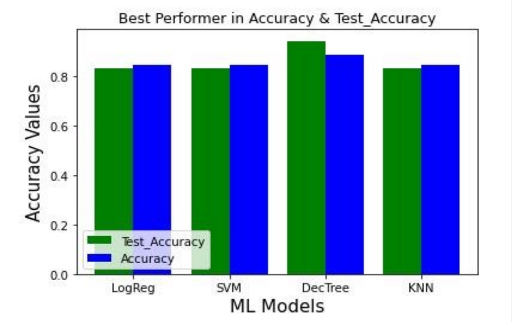
➢ Predictive Analysis:

- the best method performs in the prediction is the Decision Tree with an accuracy of 94.44%.





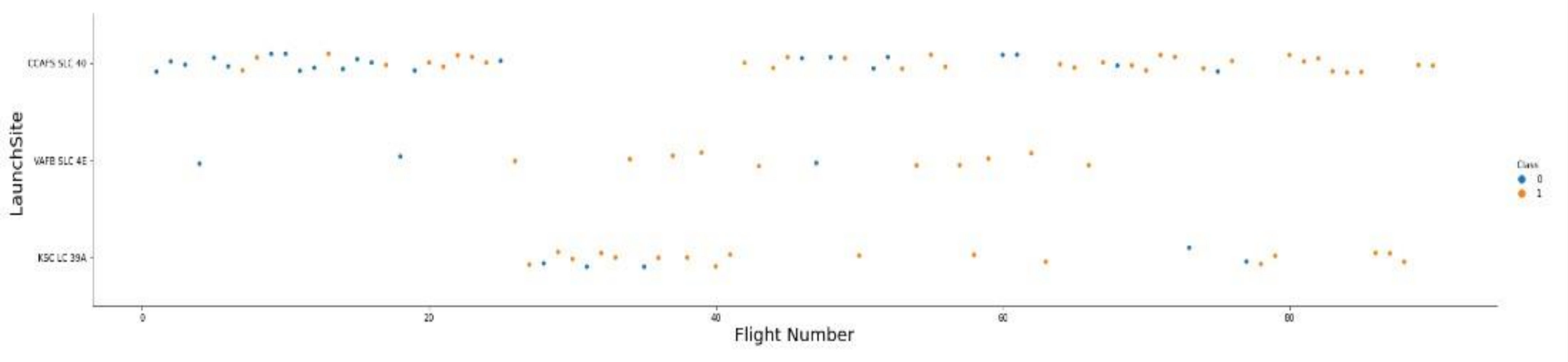




Section 2

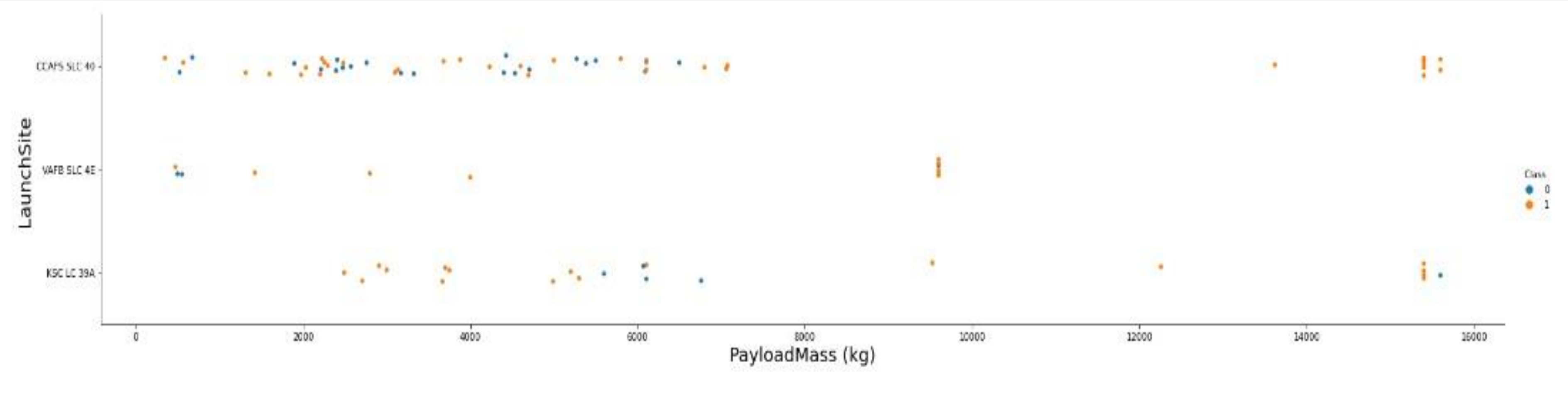
Insights drawn from EDA

Flight Number vs. Launch Site



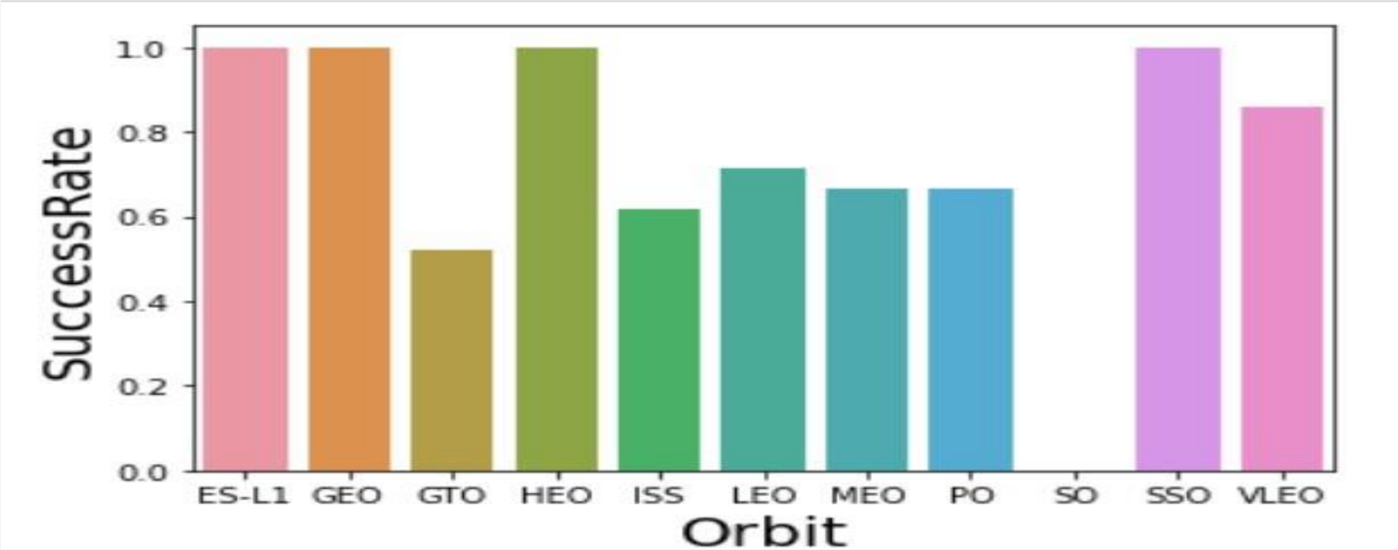
- From the Scatter Plot, its clear that there are more successful landings as the flight number increased.
- The launch site CCAFS SLC40 had a greater number of successful landing attempts and VAFB SLC 4E has a smaller number of attempts.

Payload vs Launch Site



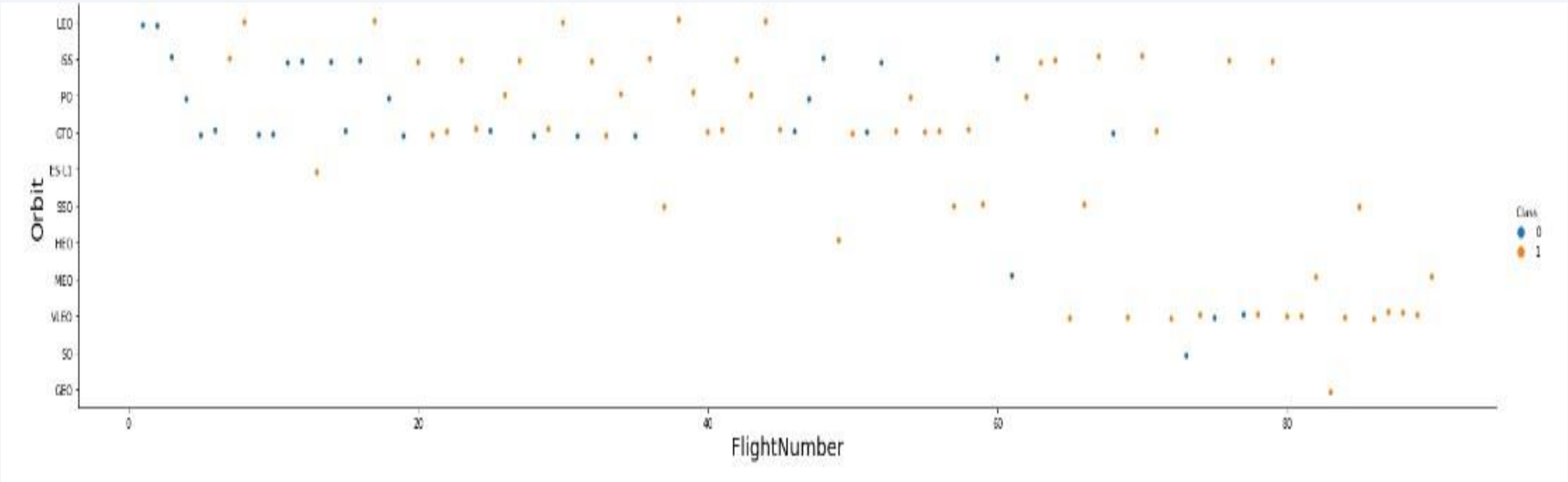
- From this Scatter plot, the greater the payload mass for CCAFS SLC 40 launch site higher the success rate.
- VAFB SLC 4E has no rockets launched for heavy payload mass above 10000kg.

Success Rate vs. Orbit Type



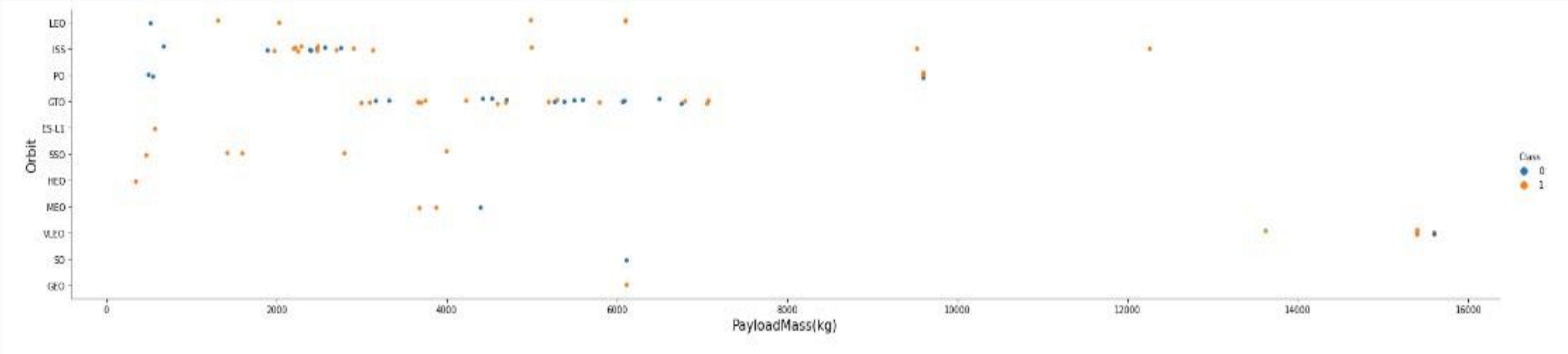
- From this Bar chart, we can understand that the Orbits ES-L1, GEO, HEO, SSO and VLEO have the most success rate.

Flight Number vs. Orbit Type



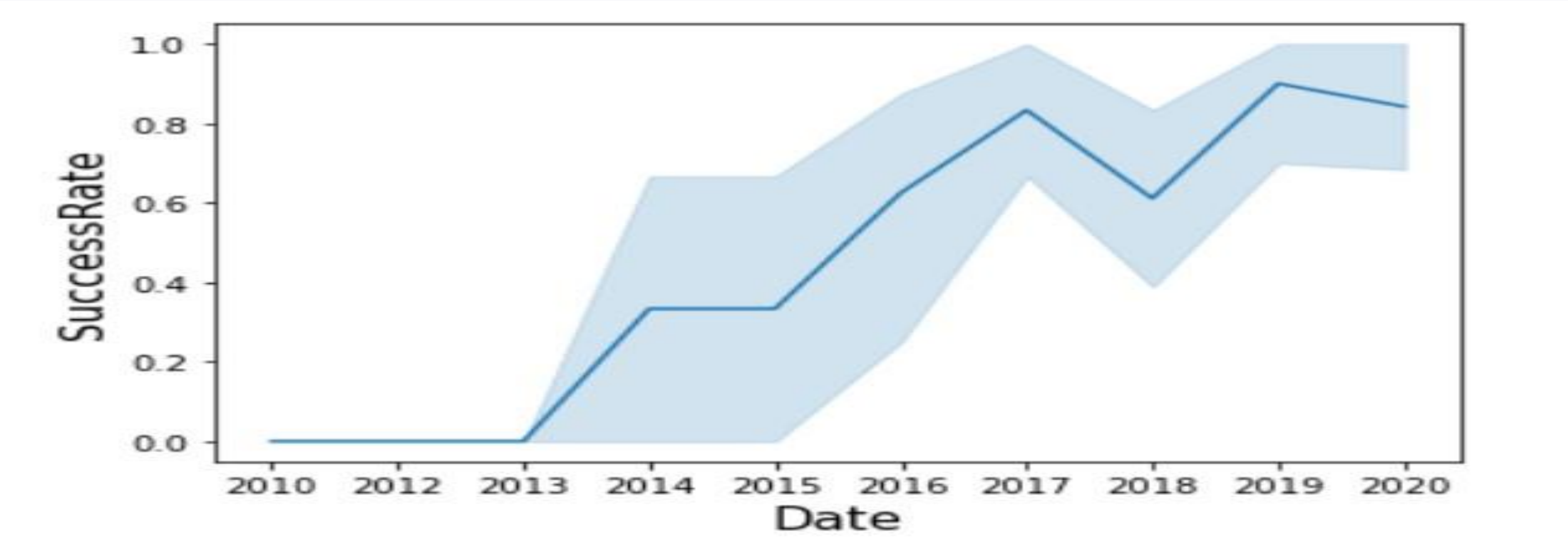
- In this Scatter Plot, we can see that LEO orbit success is related to the number of flights.
- Whereas there is no relationship between flight numbers in GTO orbit.

Payload vs. Orbit Type



- From this Scatter Plot, with heavy payloads the successful landing rate is more for POLAR, LEO, and ISS orbits.
- But for GTO orbit, we cannot distinguish as both numbers of successful and unsuccessful missions are same.

Launch Success Yearly Trend



- From the line chart it's clear that there is an increase in success rate from the year 2013 up to 2020.

All Launch Site Names

- Exploratory Data Analysis with SQL Query used for displaying the Launch Site names.

- Used the keyword DISTINCT for displaying the unique names of all launch sites from SpaceX data.

- Launch Sites names are:
  - CCAFS LC-40
  - CCAFS SLC-40
  - KSC LC-39A
  - VAFB SLC-4E

```
%sql SELECT DISTINCT LAUNCH_SITE FROM SPACEXTBL;
```

Launch Site Names Begin with 'CCA'

```
%%sql SELECT* FROM SPACEXTBL
WHERE LAUNCH_SITE LIKE 'CCA%'
LIMIT 5;
```

```
 * ibm_db_sa://mgz81488:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

- Used the SQL keyword LIKE ‘CCA%’ and LIMIT keyword(for the number) to get the records starting with the string CCA.
- Displayed the organizations launching rockets other than Space X.

Total Payload Mass

```
%%sql
SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE CUSTOMER = 'NASA (CRS)';
```

```
 * ibm_db_sa://mgz81488:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kc
Done.
```

1
45596

- Used the keyword SUM for calculating the total payload carried by boosters from NASA.

Average Payload Mass by F9 v1.1

```
%%sql
SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

```
 * ibm_db_sa://mgz81488:***@824dfd4d-99de-440d
Done.
```

1
2928

- Used the keyword AVG for calculating the average Payload mass for the Booster version F9 v1.1.
- The Average payload mass is 2928kg.

First Successful Ground Landing Date

```
%%sql SELECT MIN(DATE) FROM SPACEXTBL
WHERE LANDING__OUTCOME = 'Success (ground pad)';
```

```
 * ibm_db_sa://mgz81488:***@824dfd4d-99de-440d-999
Done.
```

1
2015-12-22

- ➢ We used SQL keyword MIN for displaying the date of the first successful ground landing.

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%%sql SELECT BOOSTER_VERSION,PAYLOAD_MASS__KG_,LANDING__OUTCOME FROM SPACEXTBL
WHERE LANDING__OUTCOME = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;
```

```
 * ibm_db_sa://mgz81488:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databas
Done.
```

booster_version	payload_mass__kg_	landing__outcome
F9 FT B1022	4696	Success (drone ship)
F9 FT B1026	4600	Success (drone ship)
F9 FT B1021.2	5300	Success (drone ship)
F9 FT B1031.2	5200	Success (drone ship)

- For displaying the names of boosters which have successfully landed on drone ship and we used SQL keyword BETWEEN for selecting payload mass greater than 4000kg but less than 6000kg

Total Number of Successful and Failure Mission Outcomes

```
%%sql SELECT MISSION_OUTCOME, COUNT(MISSION_OUTCOME) AS Total_Num FROM SPACEXTBL
GROUP BY MISSION_OUTCOME;
```

```
 * ibm_db_sa://mgz81488:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8
Done.
```

mission_outcome	total_num
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

- Used the keyword **COUNT** for calculating the total number of successful and failed mission outcomes
- Used keyword **GROUPBY** for grouping the result based on the Mission_Outcome attribute.
- In this we can understand that the missions are generally successful, only 1 mission failed

Boosters Carried Maximum Payload

```
%%sql SELECT DISTINCT BOOSTER_VERSION, PAYLOAD_MASS__KG_ FROM SPACEXTBL
WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

```
 * ibm_db_sa://mgz81488:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08
Done.
```

booster_version	payload_mass__kg_
F9 B5 B1048.4	15600
F9 B5 B1048.5	15600
F9 B5 B1049.4	15600
F9 B5 B1049.5	15600
F9 B5 B1049.7	15600
F9 B5 B1051.3	15600
F9 B5 B1051.4	15600
F9 B5 B1051.6	15600
F9 B5 B1056.4	15600
F9 B5 B1058.3	15600
F9 B5 B1060.2	15600
F9 B5 B1060.3	15600

- Used the keyword DISTINCT for the names of the booster and the keyword MAX for selecting the maximum payload mass.
- The query result shows the 12 similar booster version names, which may be of the same manufacturer carried 15600kg.

2015 Launch Records

```
%%sql SELECT LANDING__OUTCOME ,BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL
WHERE LANDING__OUTCOME = 'Failure (drone ship)' AND YEAR(DATE) = '2015';
```

```
 * ibm_db_sa://mgz81488:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08k
Done.
```

landing__outcome	booster_version	launch_site
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

- SQL query keyword **WHERE** & **AND YEAR** is used to filter records that are failed landing outcomes in drone ships, their Booster versions for the year 2015.

Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

```
%%sql SELECT LANDING__OUTCOME, COUNT(LANDING__OUTCOME) AS RANK FROM SPACEXTBL
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY LANDING__OUTCOME
ORDER BY RANK DESC;
```

```
 * ibm_db_sa://mgz81488:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1
Done.
```

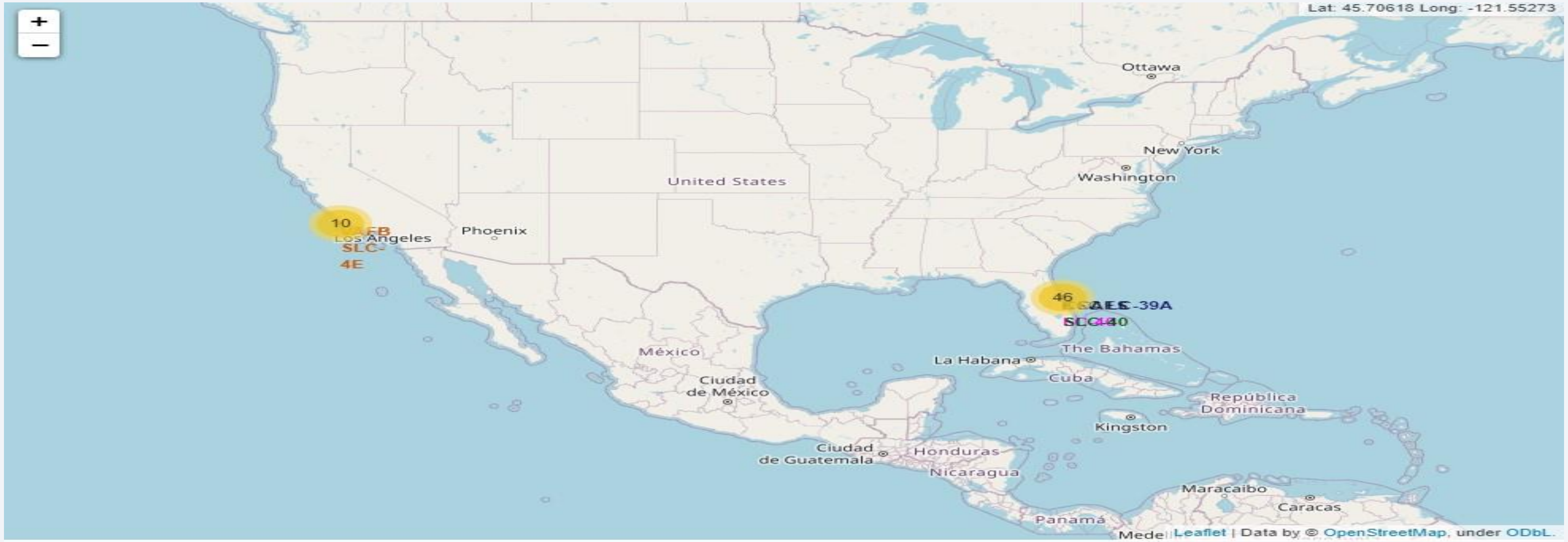
landing__outcome	RANK
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

- Combination of SQL query keyword COUNT for the number of landing outcomes, WHERE keyword to filter the launch records BETWEEN 2010-06-04 & 2017-03-20.
- GROUP BY keyword for grouping the landing outcomes, ORDER BY to order the grouped landing outcomes based on rank in descending order (DESC keyword).

Section 3

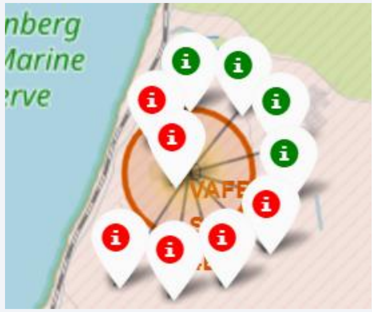
Launch Sites Proximities Analysis

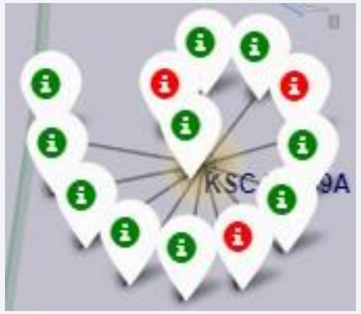
Folium Markers of Launch Sites



- All the launch sites are marked in the Folium Map with different colored folium circles.
- All launch sites are very close proximity to coastlines and a couple of thousand kilometers away from the equator line.
- 3 launch sites are concentrated in Orlando of Florida and 1 launch site in California

Launch site Markers with Color labels





VAFB SLC-4E
California Launch site

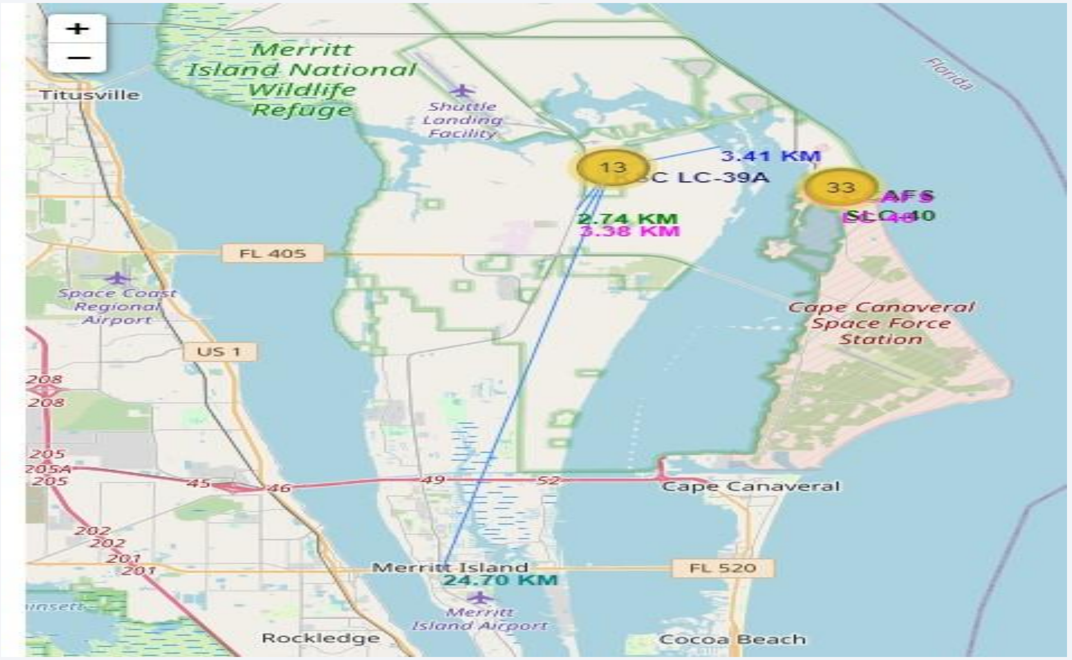
KSC LC-39A

CCAFS LC-40
Florida Launch sites

CCAFS SLC-40
Florida Launch sites

- California Launch site and Florida Launch sites are marked in Folium Map with color labels.
- GREEN marker denotes successful launches and RED marker denotes failed launches.
- Launch site KSC LC-39A has the most successful launches and least successful launches for launch site CCAFS SLC-40 when compared.

Launch Site distance to Landmarks

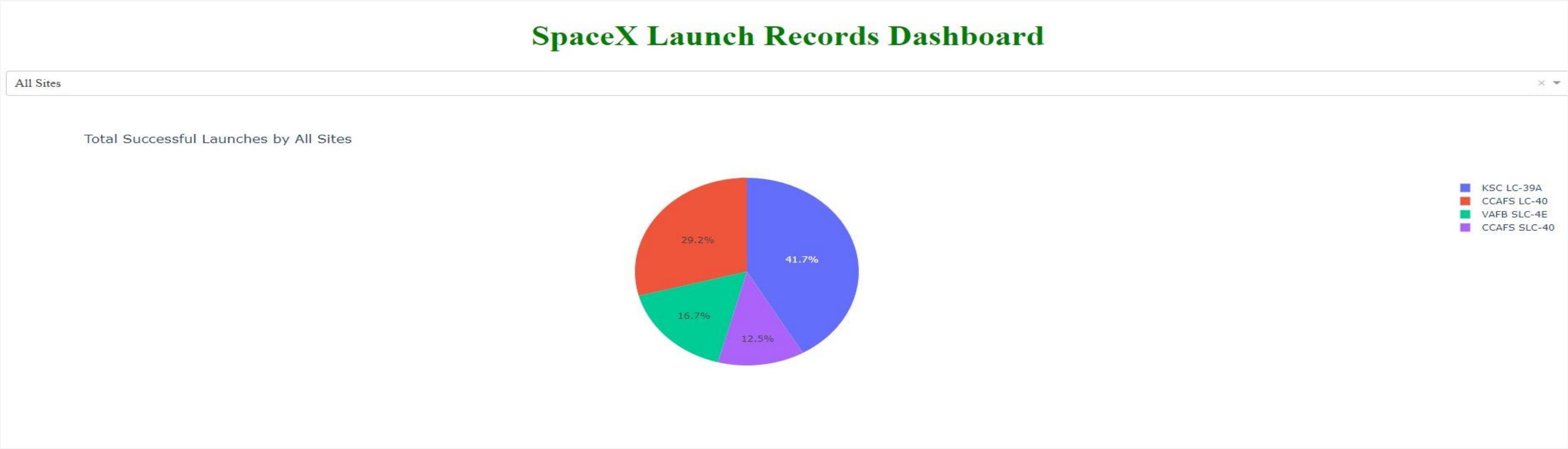


- Launch sites are usually set up a little away from cities which prevents crashes near populated areas.
- Launch sites are in very close proximity to railways and highways which helps in easy transportation of rocket parts.
- The sites are very near to the coastlines, so failed rocket launches crash into water bodies.

Section 4

Build a Dashboard with Plotly Dash

Success rate Pie chart of Launch sites

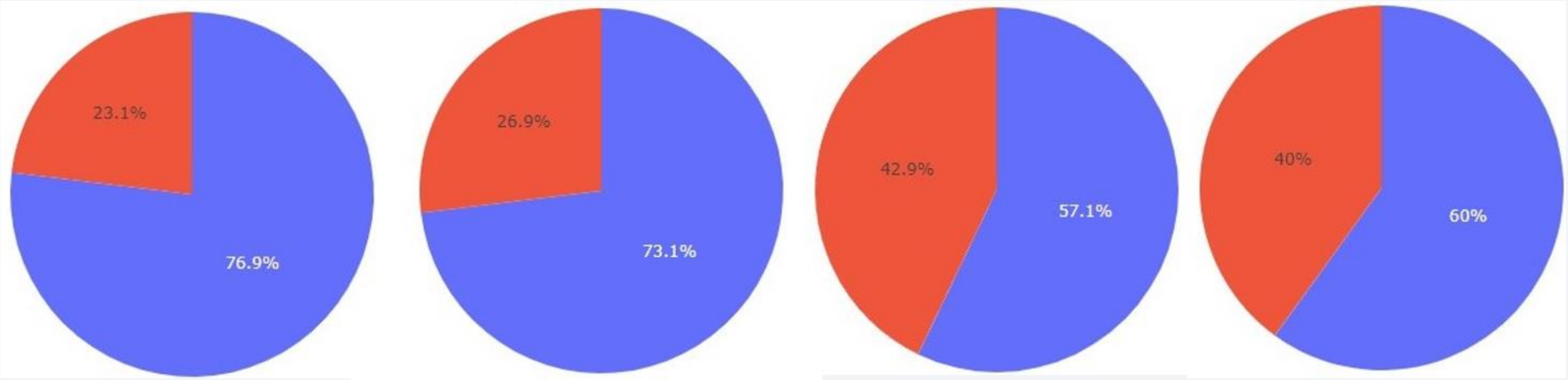


- Pie chart shows the success rate of all the launch sites
- Launch site KSC LC-39A has the most successful launches among all the other sites.

Highest launch success Pie chart

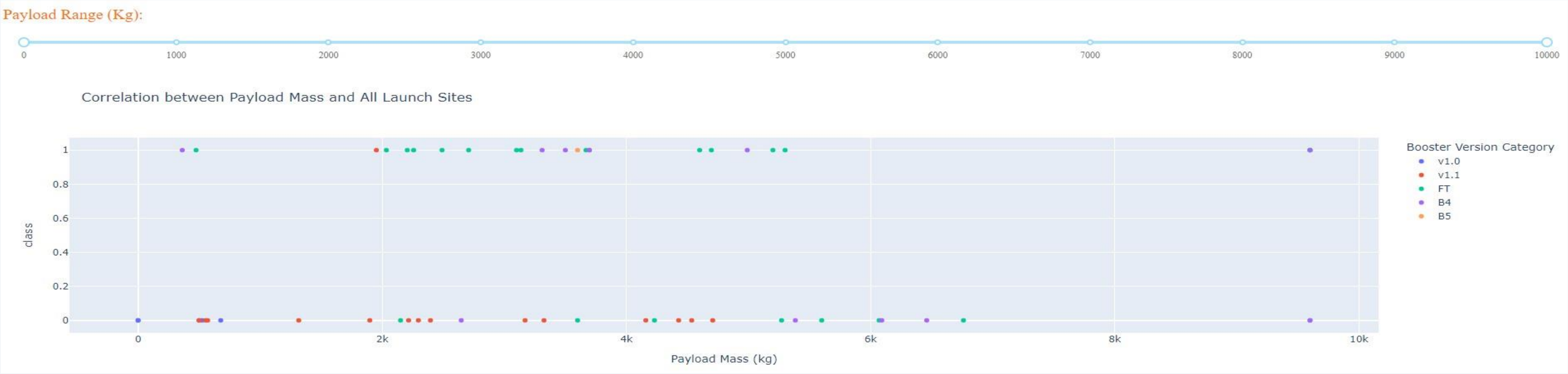
SpaceX Launch Records Dashboard

KSC LC-39A CCAFS LC-40 CCAFS SLC-40 VAFB SLC-4E



- KSC LC-39A has the highest launch success rate of 76.9%, then CCAFS LC-40 with73.1%.
- VAFB SLC-4E with 60% and least launch success rate for CCAFS SLC-40 launch site with 57.1%.

Launch success Scatter plot for All Sites

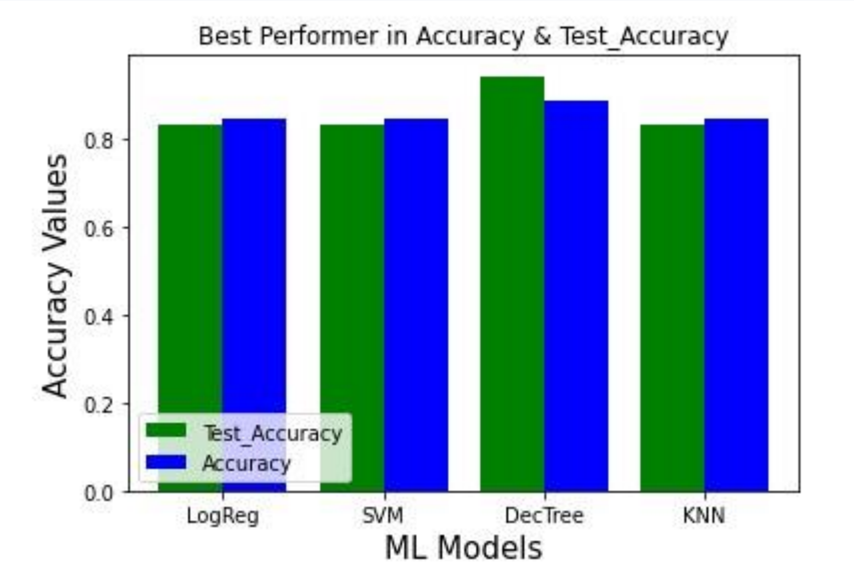


- The payload range between 2000kg and 4000kg has the highest success rate
- Booster version FT(in green color) have higher success rate than other boosters.
- Up to 5000kg payload range, booster version v1.1(red color) launches are all failed except 1 success according to the scatter plot.

Section 5

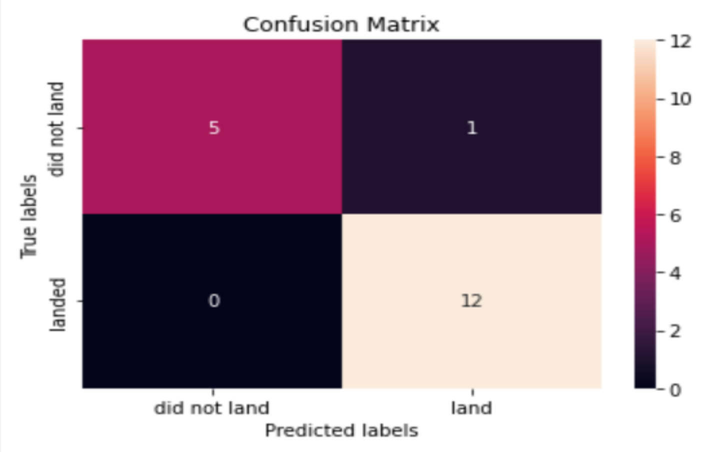
Predictive Analysis (Classification)

Classification Accuracy

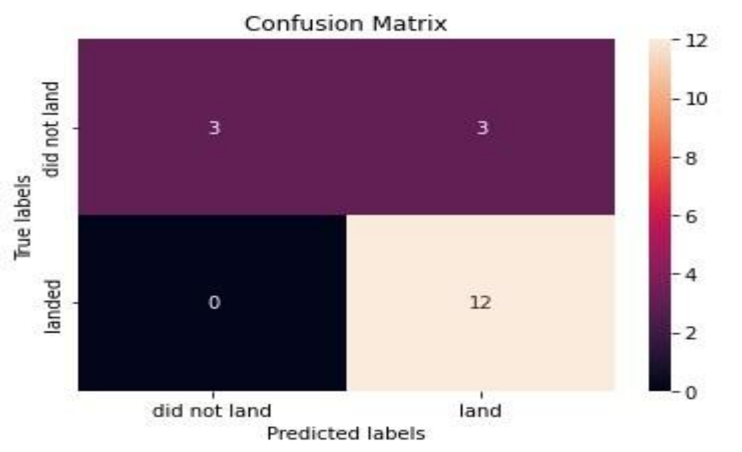


- Among all the models, Decision Tree has the highest model accuracy with 94.44%.
- The remaining 3 models have built model accuracy of 83.33%
- Bar chart visualizes that Decision Tree excels in training and testing.

Confusion Matrix



Decision Tree 94.4%



Logistic Regression, SVM & KNN 83.3%

- Confusion Matrix denotes the performance of a classification algorithm.
- Decision tree matrix on the left side identifies all 12 positive labels and 0 false negatives but failed to accurately predict 1 label(top right block).
- Matrix of Logistic Regression, SVM & KNN on the right side also identifies all 12 positive labels and 0 false negatives but failed to accurately predict 3 labels(top right block).

Conclusions

- Our goal was to use the landing information of SpaceX Falcon 9 rockets to predict the rocket launch success rate in the first stage. Through this process, the success story of SpaceX company is much clearer now.
- All the launch sites are in close proximity to railways, highways, and coastlines, which enabled easy transportation of rocket parts and prevents rockets from crashing over a populated area.
- Launch site KSC LC-39A had the highest launch success rate among the other sites.
- Launch success rate at a launch site increases with larger flight amount. The success rate started to increase from 2013 till 2020.
- Decision Tree model is the best machine learning algorithm for predicting the landing outcome of rocket launches with 94.4% accuracy.
- Predictive modeling using data science tools allows to predict the success of rocket launches at the first stage. This can be used by a company to quote a better bid than SpaceX.

Thank you!

Appendix

- All the Jupyter notebooks along with corresponding data sets are consolidated in the following repo:
- **URL**: https://github.com/itzdevisree/IBM-Data-Science-Capstone-Project/tree/master